Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

The Ego-KinoDynamic Space: Collision Avoidance for any Shape Mobile Robots with Kinematic and Dynamic Constraints

J. Minguez
jminguez@unizar.es
Dept. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, Spain

L. Montano
montano@unizar.es
Dept. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, Spain

***Abstract*— This paper presents a framework to use collision avoidance methods in the majority of existing mobile robots (that have any shape, and kinematic and dynamic constraints). The solution proposed is a vehicle abstraction layer based on transforming the space where these methods work, onto another one in which the constraints are implicitly represented. This space incorporates the vehicle kinematics and dynamics in such a way that when the reactive navigation methods are used, the motions computed comply with the motion constraints. We validate the utility of this framework by applying a classic reactive method in a real vehicle with motion constraints (whereas the original method does not address the kinematics and dynamics).**

I. Introduction

Whenever the robots must move in unknown and dynamic scenarios, sensors are required to detect and react to unforeseen obstacles. The collision avoidance methods are techniques currently used to move robots based on sensory inputs in such environments. These methods are based on a high-rate *perception-action* process, thus the sensor feedback is rapidly integrated into the framework to react to every unforeseen circumstance.

One challenge arises when these reactive methods must be used in real robots, that usually exhibit kinematic and dynamic constraints, and that have any shape. This design step has great importance in robotic technology, because ignoring the robot shape in this process inevitably leads to collisions. Furthermore, ignoring the robot kinematics and dynamics is similar to ignoring how the robot moves. This leads to prohibited motions or gross approximations in the motion, and again to collisions.

To date very few techniques address reactive collision avoidance for non-circular vehicles incorporating the kinematics and dynamics (e.g. [1] and [12]). These techniques are extensions of an existing method [5]. Thus, although good navigation results are obtained, these techniques difficulty could be re-utilized to extend other methods.

The contribution of this paper is the design of an abstraction layer of the vehicle for reactive navigation methods. The idea is to express the vehicle constraints in a space in such a way that reactive methods do not need to address them when they are used. The space construction is derived from the robot Configuration space in order to take into account the robot shape. Moreover, the space incorporates the vehicle kinematics (motions over arcs of circle) and the dynamics (the *braking distance* and the *reachability constraints*). Seen as a whole, the characteristics of the vehicle are implicitly represented in the space, and thus they are abstracted for the reactive method. Then, the advantage is that reactive navigation methods applied to this space compute motions that take into account the vehicle shape and the motion constraints (we achieve reactive collision avoidance addressing the robot shape, kinematics and dynamics with methods that do not explicitly address these constraints).

We have validated the utility of this framework with a *Potential Field Method* [6]. By using our framework, this reactive method was used to safely drive a real vehicle among locations, whereas the original method formulation does not deal with motion constraints.

II. Vehicle Shape and Motion Constraints

We focus on robots moving on a flat surface (such as two-driving wheeled robots, car-like robots, etc). Next, we discuss the shape and motion constraints.

A. Shape of the vehicle

The collision avoidance problem is usually addressed in the Workspace $\mathcal{W}$ ($\mathbb{R}^2$) if the robot shape is approximated by a circle, or in the Configuration space $\mathcal{C}$ ($\mathbb{R}^2 \times S^1$) for any shape. The research presented here is based on a spatial transformation, prior to the reactive method usage. We will demonstrate that this transformation can be applied to both spaces, thus: (i) allowing us to take into account any robot shape, and (ii) giving generality to the framework since the majority of approaches apply to these spaces.

B. Kinematics of the vehicle

The kinematic model of the robots considered here can be expressed by (see [7]):

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} cos\theta \\ sin\theta \\ 0 \end{pmatrix} v + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} w \qquad (1)$$

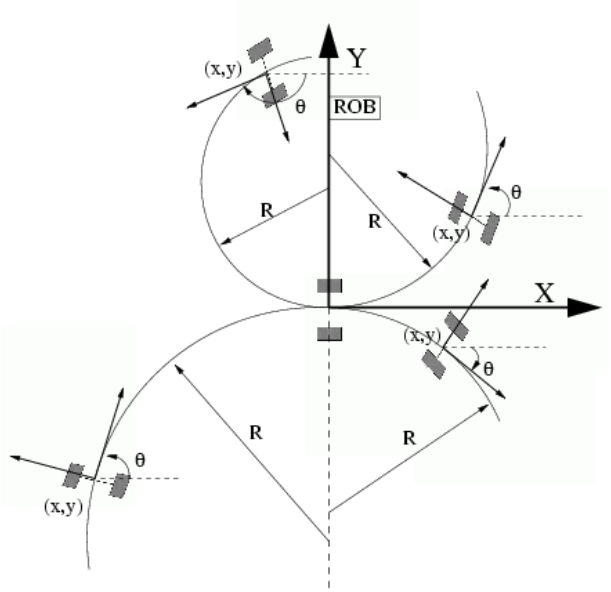

Fig. 1. The robot moves on arcs of circle or on the straight segment under the execution of a single motion command.

with v and w the linear and angular velocities.

The interest in this paper is focused on reactive navigation methods, which compute one motion command after every time interval. Under the execution of a single motion command, the vehicle paths are *arcs of circle or the straight segment* (Fig. 1). This characterization has been widely used to address the kinematic constraints (see e.g. [1], [12], [4], [5]).

The geometry of the paths

We characterize next some parameters of the robot paths, which are used in the rest of the paper. The family of admissible paths, which result from the execution of a single motion command, consists of a set of circles. In the robot frame, these circles contain the origin, and their centers (instantaneous turning center) are on the y-axis (see Fig. 1). The circle radius that leads to a location (x, y) is given by:

$$x^2 + (y - R)^2 = R^2 \quad \text{with} \quad R = \frac{x^2 + y^2}{2y} \qquad (2)$$

where $R \in]-\infty, \infty[$ is the turning radius. The robot orientation is constrained on a circular path by:

$$\theta = atan2(x, R - y) = atan2(2xy, x^2 - y^2) \qquad (3)$$

where R is given by Eq. (2), and $\theta \in [-\pi, \pi]$ is expressed in the robot reference. The distance traveled along the circumference of the circle (arc-length) to reach a location is:

$$L = \begin{cases} |x|, & y = 0 \\ |R.\theta|, & y \neq 0 \end{cases} \qquad (4)$$

where R and θ are given by Eqs. (2) and (3). We discuss next the vehicle dynamics.

C. Dynamics in Motion Commands

In reactive navigation, we are interested in motion commands that ensure: (i) the execution is collision-free during the next sample period T, and (ii) after execution, the guarantee for safely stopping the robot with an *Emergency Stop* always exists (by applying the maximum vehicle deceleration, a_v and a_w). We identify two dynamic constraints determined by the maximum acceleration/deceleration of the vehicle in this process:

1) *Braking constraint*: is the maximum distance traveled before the stop when the *Emergency Stop* is launched.
2) *Dynamic interval*: is the set of commands[1] that can be selected for motion. The *dynamic interval* is given by $v_{next} \in [v_o \pm \triangle v]$, where v_o is the current velocity. We compute $\triangle v$ by estimating the error that results from assuming that the steady state is reached instantly (the full procedure is described in [9], however we remark that $\triangle v$ depends on a_v).

III. The Ego-KinoDynamic Space

We present next the design of the vehicle abstraction layer based on a spatial representation that expresses the vehicle characteristics. The *Ego-KinoDynamic space* (*EKD-space*) results from a sequence of transformations that successively incorporate the vehicle kinematics and dynamics. These transformations are presented next .

A. The Ego-Dynamic Transformation

The original formulation of the *Ego-Dynamic Transformation* (*ED-transf*) [9] deals with robots that move in any direction (holonomic robots). We reformulate in this Section the *ED-transf* for vehicles that move on arcs of a circle. The *ED-transf* maps the Workspace ($\mathbb{R}^2$) onto the *ED-Space*, whilst incorporating the first dynamic constraint: the braking constraint (see Subsection II-C).

Let (x_{obs}, y_{obs}) be the obstacle location (see Fig 2). Let R_{obs}, θ_{obs}, and L_{obs} be the radius of the circle leading to the obstacle, the robot orientation on the obstacle, and the arc of the circle to the obstacle respectively. We analyze next each command separately:

Translational velocity (v)

The idea is to compute the location, (x^v_{safe}, y^v_{safe}), over the circle that allows the *Emergency Stop* to stop the robot at (x_{obs}, y_{obs}) (see Fig 2a). First the robot travels at a given velocity, v, during T, a distance L^v_{safe}. Next, the

[1]We analyze the case of v, but and w is analogous.

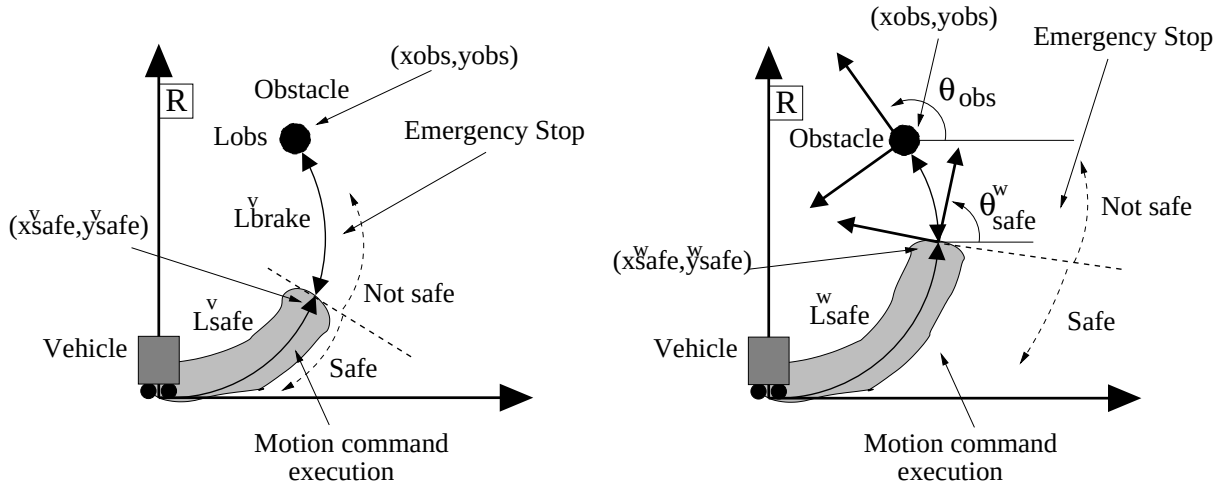

Fig. 2. (a) Translational and (b) rotational velocities case.

robot travels L^v_{brake} while braking:

$$L_{obs} = L^v_{safe} + L^v_{brake} \quad (5)$$

$$L^v_{safe} = v.T \qquad L^v_{brake} = \frac{v^2}{2.a_v} \quad (6)$$

$$\frac{(L^v_{safe})^2}{2.a_v.T^2} + L^v_{safe} - L_{obs} = 0 \quad (7)$$

$$L^v_{safe} = a_v.T^2.(\sqrt{1 + \frac{2.L_{obs}}{a_v.T^2}} - 1) \quad (8)$$

Then, (x^v_{safe}, y^v_{safe}) is the location on the circle at a distance L^v_{safe}:

$$x^v_{safe} = \begin{cases} sign(x_{obs}).R_{obs}.\sin(L^v_{safe}/R_{obs}), & R_{obs} \neq \infty \\ sign(x_{obs}).L^v_{safe}, & R_{obs} = \infty \end{cases} \quad (9)$$

$$y^v_{safe} = \begin{cases} R_{obs}.(1 - \cos(L^v_{safe}/R_{obs})), & R_{obs} \neq \infty \\ 0, & R_{obs} = \infty \end{cases} \quad (10)$$

Rotational velocity (w)

The case of the rotational velocity, w, is analogous to v, however the distances are now angle increments (see Fig. 2b). Then, θ^w_{safe} is the increment of angle, that allows the *Emergency Stop* to stop the rotational velocity (with a angle increment of θ^w_{brake}). Then:

$$\theta_{obs} = \theta^w_{safe} + \theta^w_{brake} \quad (11)$$

$$\theta^w_{safe} = w.T \qquad \theta^w_{brake} = \frac{w^2}{2.a_w} \quad (12)$$

$$\frac{(\theta^w_{safe})^2}{2.a_w.T^2} + \theta^w_{safe} - \theta_{obs} = 0 \quad (13)$$

$$\theta^w_{safe} = sign(\theta_{obs}).a_w.T^2.(\sqrt{1 + \frac{2.|\theta_{obs}|}{a_w.T^2}} - 1) \quad (14)$$

Then, (x^w_{safe}, y^w_{safe}) is the location on the circle where the robot orientation is θ^w_{safe}:

$$x^w_{safe} = \begin{cases} R_{obs}.\sin(\theta^w_{safe}), & R_{obs} \neq \infty \\ \infty, & R_{obs} = \infty \end{cases} \quad (15)$$

$$y^w_{safe} = \begin{cases} R_{obs}.(1 - \cos(\theta^w_{safe})), & R_{obs} \neq \infty \\ 0, & R_{obs} = \infty \end{cases} \quad (16)$$

The Ego-Dynamic Transformation

The locations (x^v_{safe}, y^v_{safe}) or (x^w_{safe}, y^w_{safe}) represent a motion constraint: if the vehicle travels over the circle a longer distance than $min(L^v_{safe}, L^w_{safe})$, the stop would not be possible. Thus, we select the location that corresponds to the minimum distance to build the *ED-transf*:

$$\begin{array}{rcl} \textit{ED-transf}: \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\ (x_{obs}, y_{obs}) & \rightarrow & \begin{cases} (x^v_{safe}, y^v_{safe}), & L^v_{safe} \leq L^w_{safe} \\ (x^w_{safe}, y^w_{safe}), & L^v_{safe} > L^w_{safe} \end{cases} \end{array} \quad (17)$$

Notice that any location in the resulting space depends on: (i) the location of the obstacle (x_{obs}, y_{obs}), (ii) the deceleration capabilities of the robot (a_v, a_w), and (iii) the sampling period (T) in which the motion command is applied. However the locations do not depend on the robot velocity.

The *ED-transf* maps any point of the Workspace onto the *ED-space*, while incorporating the braking constraint (that depends on the robot deceleration and the sample period). Moreover, the *ED-space* can be computed in closed form for obstacle points and the complexity of the transformation is lineal with the number of obstacle points.

Next, we present the transformation that incorporates the kinematics.

B. The Ego-Kinematic Transformation

The *Ego-Kinematic Transformation* [10] (*EK-transf*) maps any point of $\mathbb{R}^2$ to a space that we represent in polar coordinates for convenience:

$$\begin{array}{rccc} \textit{EK-transf}: & \mathbb{R}^2 & \rightarrow & \mathbb{R}_0^+ \times [-\pi, \pi] \\ & (x, y) & \rightarrow & (L, \alpha = atan2(R^{-1}, sign(x)) \end{array} \quad (18)$$

with L and R the arc length and radius of the circle leading to the point.

The *EK-transf* transforms circular paths (with radius R and arc length L) into straight paths (with direction α and length L). The interest is that each point of the resulting space (*EK-space*) is reached by a straight-line motion ("free-flying behavior"), which represents a motion over an admissible path for the robot. Furthermore, the *EK-space* can be computed in closed form with a complexity linear with the number of obstacle points, and the transformation is invertible (that is *EK-transf* $^{-1}$ exists). For more details see [10].

C. The Ego-KinoDynamic Transformation

The *Ego-KinoDynamic Transformation* (*EKD-transf*) is obtained by applying the sequence of the *ED-transf* and *EK-transf*:

$$\begin{array}{rccc} \textit{EKD-transf}: & \mathbb{R}^2 & \rightarrow & \mathbb{R}_0^+ \times [-\pi, \pi] \\ & (x, y) & \rightarrow & \textit{EK-transf}\ (\textit{ED-transf}\ (x, y)) \end{array} \quad (19)$$

The *EKD-transf* leads the obstacle information to the *EgoKinoDynamic space* (*EKD-space*). This transformation has the above-mentioned properties of the *ED-transf* and the *EK-transf*. The interest is that in the *EKD-space* the robot is free of kinematic constraints (any location is reached by a straight-line motion), and the braking constraint is represented in the space.

D. The Spatial Window in the EKD-space

Up to now, we have incorporated in the *EKD-space* the vehicle kinematics, and the first of the dynamic constraints (braking constraint). However, we still need to consider the second dynamic constraint: the selection of a dynamically admissible motion (see Subsection II-C).

Lets say that we select an arbitrary location (L_p, α_p) in the *EKD-space*, then (v, w) is computed by:

$$(x_p, y_p) = \textit{EK-transf}^{-1}(L_p, \alpha_p) \quad (20)$$

$$(v, w) = (sign(x_p).\frac{L_p}{T}, \frac{\theta_p}{T}) \quad (21)$$

where L_p and θ_p are given by Eqs. (4,3) for (x_p, y_p). However this command (v, w) could not be dynamically admissible. Then, we need to compute the locations of the *EKD-space* that come up with dynamically admissible motions. For this, we obtain the reachable locations of the Workspace by admissible motions by applying the vehicle motion equations to the commands within the *dynamic interval* for $t = T$ (sample period):

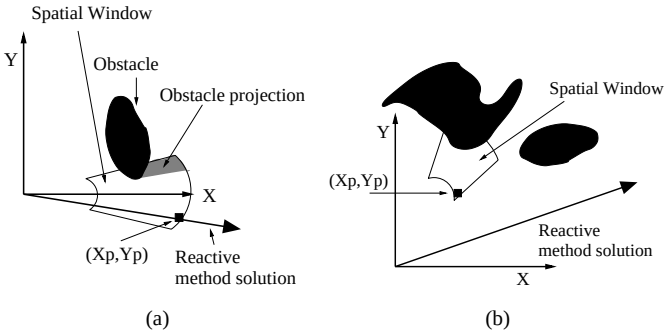

Fig. 3. Solution locations within the SW in the EKD-space (represented in Cartesian coordinates).

$$(x_{SW}, y_{SW}) = \begin{cases} (\frac{v}{w}.\sin(w.T), -\frac{v}{w}.(\cos(w.T) - 1)), & \text{if } w \neq 0 \\ (v.T, 0), & \text{if } w = 0 \end{cases} \quad (22)$$

We call this set of locations the *Spatial Window* (SW), that we transform to the *EKD-space* by applying the *EK-transf* (see Fig. 3). Any location within the SW in the *EKD-space* leads to a dynamic admissible command (computed by Eqs. (20,21). However, we need to eliminate the locations of the SW that lead to collisions. The collision locations within the SW are created either by obstacles or by the projection of the obstacles. Fig. 3a depicts the collision locations created by the obstacle within the SW and by the obstacle projection (the projection is the part of the SW that is occluded from the robot frame origin, because the robot move in straight segments in this space).

Notice that if we provide a procedure to compute a collision-free location within the SW in the *EKD-space*, this leads to a collision-free motion command that complies with the dynamics and kinematics: **the objective of this work**. We describe below the usage of reactive navigation methods to achieve this goal.

As mentioned in Section II-A, the majority of collision avoidance methods apply either to the Workspace or to the Configuration space. So far the whole procedure has been developed for the Workspace. In the following Subsection we show that the *EKD-transf* can also be applied to the robot Configuration space.

E. Applying the EKD-transf in the Configuration space

For the mobile robots addressed in this paper, the Configuration space, $\mathcal{C}$, includes both the vehicle position and orientation, i.e. $\mathcal{C} \sim \mathbb{R}^2 \times S^1$. As discussed before, these vehicles move on circular paths. On a circle, the robot orientation is constrained by Eq. (3) (see Fig. 1).

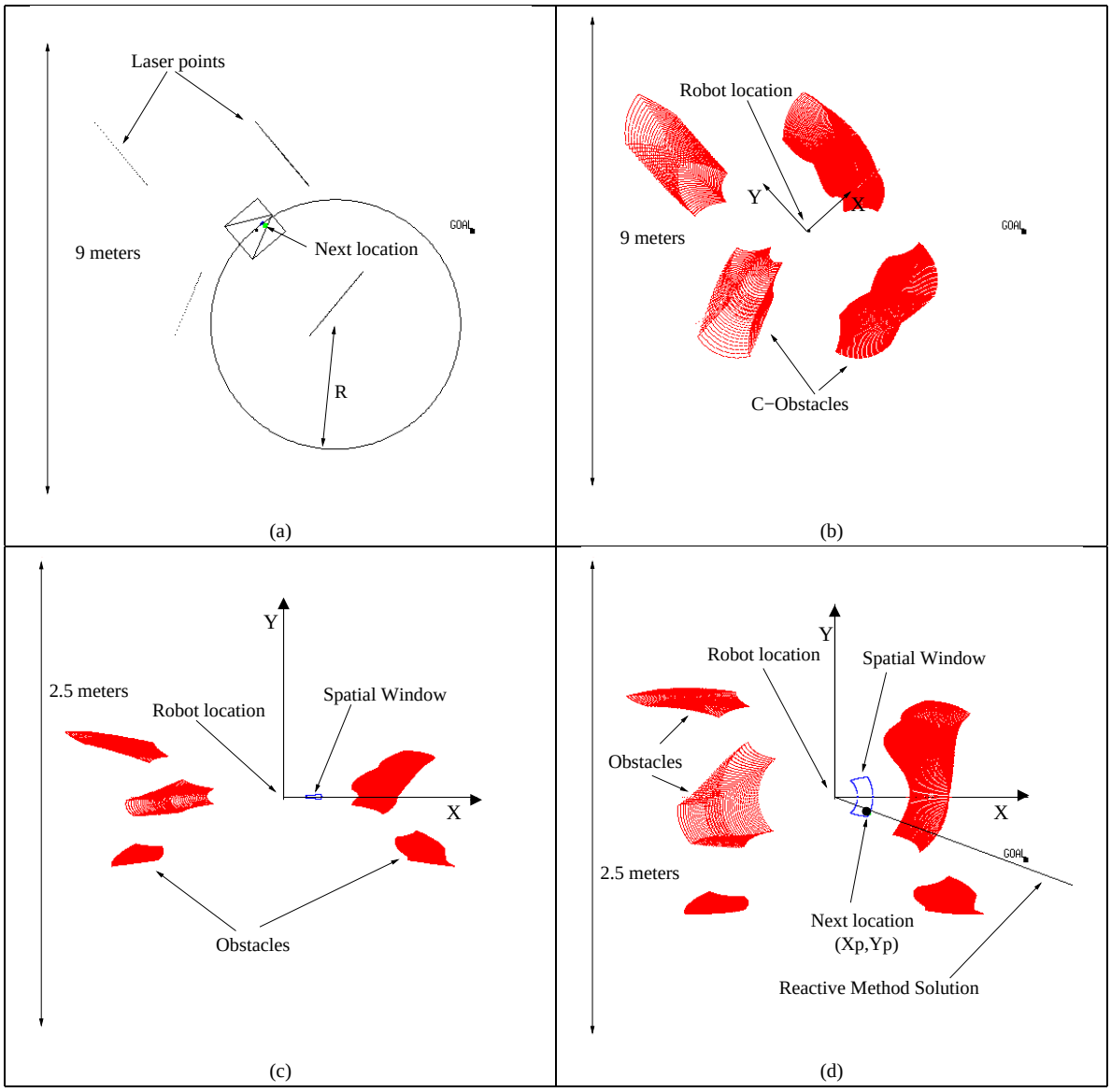

Fig. 4. a) Robot in the Workspace. b) Obstacle information in the subset of the Configuration space c) *ED-space*. d) *EKD-space* represented in Cartesian coordinates.

This Equation expresses a holonomic equality constraint. The effect is to reduce the dimension of the Configuration space by one (notice that once a location (x, y) is fixed, θ is given by Eq. (3)). Thus, the robot Workspace and the relevant subset of the Configuration space are described by $\mathbb{R}^2$. Furthermore, [10] describes an algorithm to map the sensory information (obstacles) in this subset for any robot shape (see Figs. 4a,b). Therefore, the *EKD-transf* can be used to map the obstacles of this subset of the Configuration Space in exactly the same way as it was used in the robot Workspace.

In summary, this Section has presented the *EKD-space*, where the dynamics and kinematics are implicitly represented. Besides we have discussed how the *EKD-space* can be derived from the Workspace or from the Configuration space. This gives generality to the framework, whilst also giving the possibility of embedding the vehicle shape in the space. We address next the usage of the *EKD-space* in order to use reactive navigation methods in robots with any-shape, kinematic and dynamic constraints.

IV. Applying Reactive Navigation Methods in the Ego-KinoDynamic Space

This Section presents the reactive navigation using the *EKD-space*. To achieve this goal, we exploit that the solution of most reactive navigation methods are *most promising motion directions* (e.g. [6], [2], [8]). Then, the strategy is to apply these methods on the *EKD-space*,

and utilize the solutions to select a collision-free location within the *Spatial Window* (that fixes a motion command). The procedure at each sampling period T is:

1) The obstacle information is reduced to points[2] expressed in the robot frame of reference (Fig. 4a). Then, depending on whether the reactive method[3] applies to the Workspace or to the Configuration space:
   a) If it applies to the Workspace the *ED-transf* is applied to the obstacle points.
   b) If not, we compute the $\mathcal{C}$-Obstacle region in the two dimensional subset of the Configuration space (Fig. 4b). The *ED-transf* is applied next.

   In both cases the result is the obstacle information in the *ED-space* (Fig. 4c).
2) The *EK-transf* is applied to the obstacle information in the *ED-space*, leading the obstacle information to the *EKD-space* (Fig. 4d).
3) The reactive method is applied to the *EKD-space* to compute a direction solution, "reactive method solution" in Fig. 4d.
4) The direction solution is used to select a collision-free location, (x_p, y_p), within the SW in the *EKD-space*. Our strategy is:
   a) If the direction solution intersects the SW, the closest collision-free location to the direction solution that favors forward progress is selected (see Fig. 3a and Fig. 4d).
   b) If not, we select the closest collision-free location within the SW to both the robot location and the direction solution. This heuristic reduces the robot velocity, while bringing the SW closer to the reactive method solution (Fig. 3b).

   As a result a collision-free location (x_p, y_p) within the SW is selected (Fig. 4d).
5) The motion command (v, w) is computed following the procedure presented in Subsection III-D (Fig. 4a depicts the turning radius $R = \frac{v}{w}$).

In this framework it could be possible that the SW does not contain collision-free locations. In this case the *Emergency Stop* is launched to safely stop the robot. Subsequently, the motion is resumed.

The main interest of this framework is that the vehicle constraints are abstracted from the reactive method, since they are represented in the *EKD-space*. The reactive method is only used to select a *most promising direction* of motion in this space, which is subsequently used to compute the motion. Thus, many reactive algorithms can be used within this framework, and as a consequence, the motion takes into account the vehicle shape, kinematics and dynamics. Furthermore, the safety of the reactive navigation method is improved, since the motions are *Collision-free* and the guarantee for safely stopping the robot always exits. We present in the next Section experimental results.

[2]Notice that the obstacle information in reactive navigation is usually given in the form of points (sensory input).

[3]We use here a *Potential Field Method* that applies to the Configuration space, however the framework is presented in general.

V. Experimental Results

Our intention is to move a wheelchair vehicle[4] (see Fig. 5) with a Potential Field Method (PFM) [6].

Fig. 5. The wheelchair vehicle.

In this case, a circle is a coarse approximation of the vehicle shape due to the area swept when turning (notice that the tractor wheels are in the back part of the robot). The vehicle moves over arcs of circle, thus to ignore the kinematics would rely in gross approximations in the motion (putting safety at risk). To ignore the vehicle dynamics would lead to commands that cannot be executed, and then, to motions that are not the planned ones (again putting safety at risk).

The challenge here is that this reactive navigation method (PFM) does not consider the vehicle kinematics and dynamics. Then, to overcome these difficulties, we use the *EKD-space* framework to abstract the vehicle from the reactive navigation method. Therefore, we can use the PFM on the vehicle while taking into account the vehicle shape and all the motion constraints.

Fig. 6a shows a experiment in a scenario where a human was randomly placing obstacles around. The robot successfully avoided the unforeseen obstacles while moving towards the goal location (the only information given in advance).

During all the experiment, the motion commands computed complied with the vehicle kinematics since motions on circular paths are taken into account in the framework, and thus (v, w) are computed (notice that arcs of circle mainly compose the trajectory carried out in Fig. 6a, and the motion commands computed in Figs. 6b,c). The reference commands were dynamically admissible for the vehicle, because they were always computed within the *dynamic interval*. As a consequence, the vehicle could execute the motion planned (see the reference commands and the controller behavior of the vehicle Figs. 6b,c). Furthermore, the performance of the PFM method was improved, because the guarantee for safely stopping the robot if required always existed (as the braking constraint

[4]Differential-driven robot equipped with a 2D laser sensor.

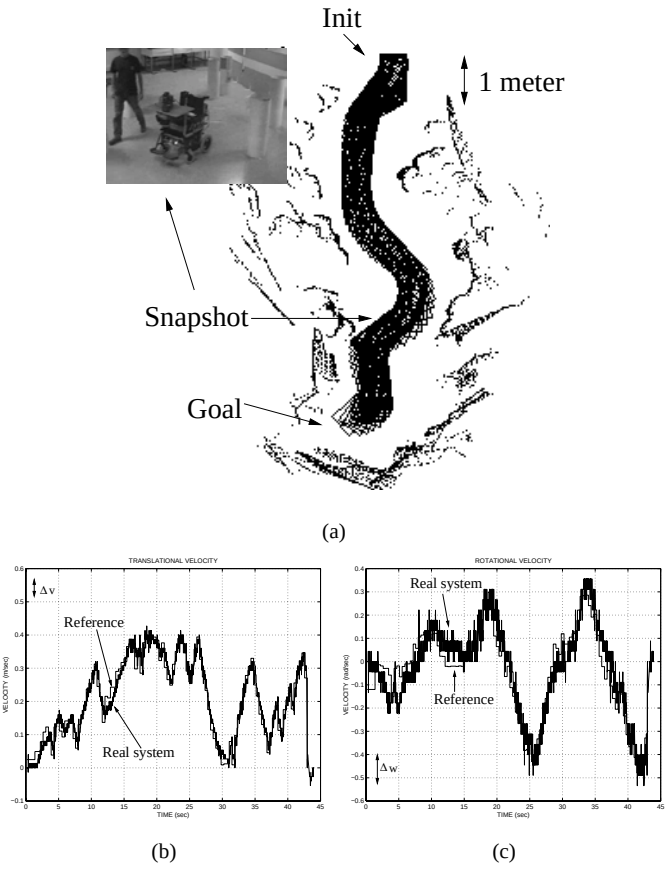

Fig. 6. (a) Experiment with the real vehicle. (b) and (c) Reference commands (v, w) and controller behavior.

is taken into account). We remark that the vehicle shape was also taken into account because the PFM is defined in the Configuration space, and thus we construct the *EKD-space* from the relevant subset of the Configuration space (the procedure is illustrated step by step in Fig. 4).

VI. Conclusion

We have presented in this paper a framework that abstracts the vehicle constraints from reactive methods. This allows applying many existing collision avoidance methods (that do not consider these constraints its basic formulation) to many of the existing robots. We have validated this framework by applying an existing method for collision avoidance (PFM) to a real vehicle addressing the motion constraints (whereas the original method does not address the motion constrains). These experiments have been carried out for a two-wheeled robot, but it could also be used on tri-cycle robots and car-like robots (see [10]).

The usage of this framework does not avoid the local nature of reactive navigation methods: the trap situations and the cyclic behaviors in the robot motion persist. However, our framework could be used together with techniques that aim to increase the locality of reactive methods, such as those described in [14], [3], [11], [13]. Then, these undesirable situations would be mitigated.

VII. Acknowledgements

This work was partially supported by MCYT DPI2000-1272.

VIII. References

[1] K. Arras, J. Persson, N. Tomatis, and R. Siegwart. Real-time Obstacle Avoidance for Polygonal Robots with a Reduced Dynamic Window. In *IEEE Int. Conf. on Robotics and Automation*, pages 3050–3055, USA, 2002.

[2] J. Borenstein and Y. Koren. The Vector Field Histogram–Fast Obstacle Avoidance for Mobile Robots. *IEEE Transactions on Robotics and Automation*, 7:278–288, 1991.

[3] O. Brock and O. Khatib. High-Speed Navigation Using the Global Dynamic Window Approach. In *IEEE Int. Conf. on Robotics and Automation*, pages 341–346, Detroit, MI, 1999.

[4] W. Feiten, R. Bauer, and G. Lawitzky. Robust Obstacle Avoidance in Unknown and Cramped Environments. In *IEEE Int. Conf. on Robotics and Automation*, pages 2412–2417, 1994.

[5] D. Fox, W. Burgard, and S. Thrun. The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, 4(1), 1997.

[6] O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. *Int. Journal of Robotics Research*, 5:90–98, 1986.

[7] J. Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. *Robot Motion Planning and Control*, 229, 1998.

[8] J. Minguez and L. Montano. Nearness Diagram Navigation (ND): A New Real-Time Collision Avoidance Approach. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2094–2100, Takamatsu, Japan, 2000.

[9] J. Minguez, L. Montano, and O. Khatib. Reactive Collision Avoidance for Navigation at High Speeds or Systems with Slow Dynamics. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 588–594, Switzerland, 2002.

[10] J. Minguez, L. Montano, and J. Santos-Victor. Reactive Collision Avoidance for Non-holonomic Robots using the Ego-Kinematic Space. In *IEEE Int. Conf. on Robotics and Automation*, pages 3074–3080, Washington, USA, 2002.

[11] J. Minguez, L. Montano, N. Simeon, and R. Alami. Global Nearness Diagram Navigation (GND). In *IEEE Int. Conf. on Robotics and Automation*, pages 33–39, Seoul, Korea, 2001.

[12] C. Schlegel. Fast Local Obstacle Avoidance under Kinematic and Dynamic Constraints for a Mobile Robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Canada, 1998.

[13] C. Stachniss and W. Burgard. An Integrated Approach to Goal-directed Obstacle Avoidance under Dynamic Constraints for Dynamic Environments. In *IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, pages 508–513, Switzerland, 2002.

[14] I. Ulrich and J. Borenstein. VFH*: Local Obstacle Avoidance with Look-Ahead Verification. In *IEEE Int. Conf. on Robotics and Automation*, pages 2505–2511, 2000.